REPRESENTING PROCEDURAL KNOWLEDGE IN EXPERT SYSTEMS- AN APPLICATION TO PROCESS CONTROL

Massimo Gallanti (*), Giovanni Guida (&), Luca. Spampinato (+), Alberto Stefanini (*)

(♦) CISE, P.O. Box 12081, 20134 Milano, Italy
(&) Politecnico di Milano, Artificial Intelligence Project, Milano, Italy
(+) Quinary, Milano, Italy

ABSTRACT

The paper presents a novel expert system architecture which supports explicit representation and effective use of both declarative and procedural knowledge. These two types of expert knowledge are represented by means of production rules and event-graphs respectively, and they are processed by a unified inference engine. Communication between the rule level and the event-graph level is based on a full visibility of each level on the internal state of the other, and it is structured in such a way as to allow each level to exert control on the other.

This structure offers several advantages over more traditional architectures. Knowledge representation is more natural and transparent; knowledge acquisition turns out to be easier as pieces of knowledge can be immediately represented without the need of complex transformation and restructuring; inference is more effective due to reduced non-determinism resulting from explicit representation of fragments of procedural knowledge in event-graphs; finally, explanations are more natural and understandable.

The proposed architecture has been adopted for the design of PROP, an expert system for on-line monitoring of the cycle water pollution in a thermal power plant. PROP is running on a SUN-2 workstation and has been tested on a sample of real cases.

INTRODUCTION

The artificial intelligence community has been successfully applying, since the mid-seventies, new powerful rule-based techniques to applications that could not be faced with more traditional approaches. In fact, the complexity of any classical solution was so evident as to discourage their use from the beginning.

Rule-based system technology (Hayes-Roth, Waterman, and Lenat, 1983) has offered a basic advantage over traditional programming: it has enabled the programmer to tackle problem solving at a higher level of abstraction and in a more flexible and natural way. The usual activities of problem analysis, invention of a solution algorithm, and programming are replaced in the expert system approach by the representation of knowledge about the application domain (including available resources, constraints, and problem solving skills), of the specific problem to be solved, and of the goal to be achieved. Responsibility about *how* to use knowledge in order to solve the problem is left to the system, and the programmer only has to represent *what* is known about the problem and is likely to be relevant to its solution. The new concept of knowledge-based problem solving appears, therefore, as another way of programming, suitable to large classes of challenging application domains, where, on the other hand, usual programming methods fail.

A major reason for the success of expert system technology is the way knowledge on the problem domain can be represented to the system. This offers in fact several highly desirable features, such as:

- ability to represent knowledge in a highly declarative way, without bothering about its use;
- possibility of describing fragmentary, ill-structured, approximate, uncertain, heuristic knowledge, that is often of crucial importance in applications;
- possibility of incrementally creating, debugging, and updating very large knowledge bases.

These features have made rule-based systems a real success In all those application domains where knowledge involved is mainly declarative and fragmentary in nature (Buchanan and Duda, 1983).

The wave of such successes, as well as increasing interest in the industrial world, has led artificial intelligence researchers to tackle new classes of problems where, in addition to declarative and fragmentary knowledge, a lot of well structured chunks of procedural knowledge is naturally available and has to be represented and used (Friedland, 1981). Consider, for example, such areas as decision making, fault detection and diagnosis of complex systems, sensory data Interpretation, monitoring of industrial processes, etc.

In these cases, it would be foolish to disperse the available procedural knowledge Into a flat declarative representation. In fact, this would be costly and heavy with respect to knowledge acquisition, as It implies complex fragmentation and coding of knowledge. Moreover, It would cause a serious overload of the inference mechanism with the additional task of reconstructing, through a heavy non-deterministic search process, the correct procedural constraints between scattered fragments of knowledge. Finally, it would make the explanation mechanism of the inference engine very oompiex, quite unnatural, and practically useless.

It is evident that, for a number of applications where the classical rule-based system paradigm is not fully adequate, a new mechanism is needed, which can combine the structured coding of procedural knowledge proper of traditional programming with the declarative representation typical of a pure rule-based system.

The issue of representing and using procedural knowledge in rule-based systems has been widely addressed in recent years in the frame of the studies about meta-knowledge (Davis, 1980a, 1980b; Georgeff, 1982; Genesereth, 1983; D'Angeio, Guida, Pighin, and Tasso, 1985). But the focus of attention is here more on the procedural aspect of control knowledge rather than on a fusion of procedural and declarative representations both at meta-level and domain level.

This topic has been only recently addressed in a neat way in

(+) Also with Universita* di Milano, Isituto di Cibernetica, Milano, Italy

This research has been supported by ENEL-DSR (Ente Nazionale per 1'Energia Elettrioa - Diresione Studi e Ricerche).

(Georgeff and Bonollo, 1983). Their approach combines explicit representation of fragments of procedural knowledge with a mechanism for pattern-directed invocation (both goal-directed and data-driven). This approach allows effective solution of a class of practical problems, specially of consultation type, and discloses the challenging issue of heterogeneous knowledge representation in the design of expert system architectures.

The purpose of the work reported in this paper is the development and experimentation of a new approach to the integration of rule-based mechanism and computation according to procedural knowledge, that can meet the following goals:

effective and natural representation of pieces of procedural knowledge by means of a powerful and general language;

use of production rules for representing declarative knowledge;

integration of the two components into a unitary inference mechanism.

This research effort yielded to the proposal of a novel expert system architecture that has been tested in the design and implementation of PROP, an expert system for malfunction diagnosis and process surveillance concerning on-line monitoring of water pollution in a thermal power plant.

The purpose of this paper is to introduce and discuss the main features of the new proposed architecture and to illustrate the results obtained in a practical application.
The general organization of the proposed expert system paradigm is first introduced and compared to related works. Afterwards, the application addressed is briefly described, and the specific architecture of the implemented expert system is illustrated. Examples of PROP operations are presented. Finally, research results are critically evaluated and possible extensions of the proposed approach are outlined.

PRODUCTION RULES AND EVENT GRAPHS

The general architecture of the novel expert system paradigm we have developed is shown in Figure 1.
It comprises two knowledge bases, namely a *rule base* and an *event-graph, base.* The former is used to contain declarative knowledge expressed by means of usual production rules, while the latter embodies procedural knowledge represented through event-graphs.

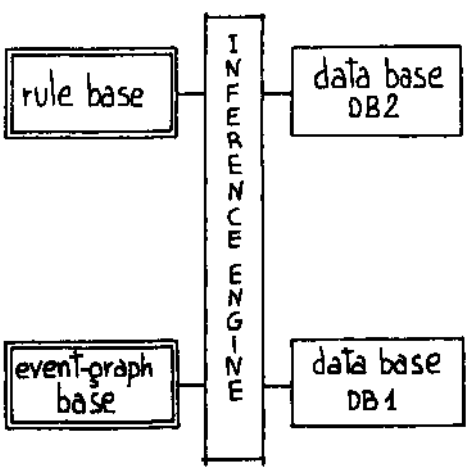

Figure 1 - Basic system architecture

The notion of event-graph shares several features with the Petri net formalism (Reisig, 1982).
We disregard here the formal definition of an event-graph and only focus on its basic expressive features.
An *event-graph (EG)* is a directed graph with two types of nodes, namely *places* and *events,* and a marking concept. Nodes termed places are graphically represented as circles and nodes termed events are graphically represented as boxes. Directed arcs can only connect nodes of different type. Events are labeled with two expressions, specifying a condition and an *action* respectively. A sample EG is shown in Figure 2.

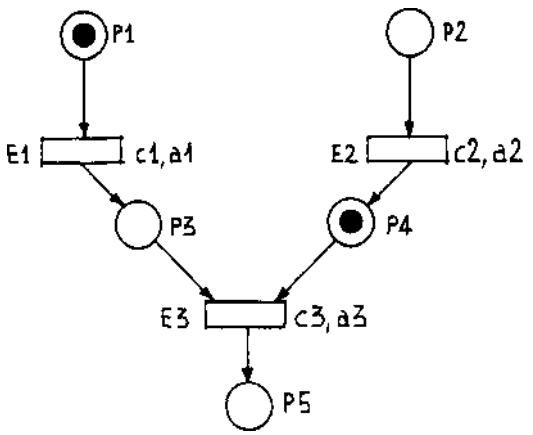

Figure *2* - A sample event-graph

A place can contain a *mark;* an EG can have an arbitrary number of marks: the set of all marked plaoes defines the *current marking* of the EG. For example, the current marking of the EG in Figure 2 is {P1, P4}. An *initial marking* is defined for each EG.

An event is *enabled* if all its ancestor places, called the *input places, are* marked. An event can fire, i.e. it is *active,* if it is enabled and its condition is true.
We define as *current conditions* of an EG the set of conditions labeling the enabled events. For example, the current conditions of the EG in Figure 2 is {c1}. An EG is *active,* (resp., *enabled)* if at least one of its events is active (resp., enabled). *Firing* an event means unmarking its input places and marking all its successor places, called the *output places.* Firing an event also causes the execution of the action specified in the event. Current marking thus evolves by means of event firing.
We stress that the concept of event-graph embodies a static and a dynamic part. The static part, i.e. the definition of places, arcs, and events with conditions and actions (plus the initial marking), represents the code of a chunk of procedural knowledge. The dynamic behaviour of an event-graph is represented by the sequence of current markings that the event-graph goes through as a consequence of successive event firing. Thus, the static part of an event-graph may be aotivated in different contexts and produce several Images corresponding to different current markings, similar to a re-entrant procedure which can be executed several times in parallel with different parameters.

Two *data bases, DB1* and *DBS,* are available to the system to be used by EGs and rules. They contain knowledge on the current state of the inference process and also constitute the link between the system and the external world (user, environment, etc.). as they can accept input data and produce output messages. Data bases DB1 and DB2 are first

initialized with information coming from outside (initial problem description). A modified recognize-act cycle is then entered, which is controlled by the achievement of a goal condition. This cycle comprises two phases, (see Figure 3}

[1] event-graph level phase:
EGs are first matched with the content of DBI. The set of enabled EGs is determined and conflict resolution is performed. The current conditions of selected (enabled) events of EGs are then matched with the content of DBI and the set of active EGs is determined. Conflict resolution is performed again. All selected (active) events are eventually fired in parallel, and the corresponding actions are executed on DBI and DB2;

[2] rule level phase:
rules are first matched with the content of DB2. The set of active rules is then determined and conflict resolution is performed. Selected rule/s is/are eventually executed on DB2 and DBI.

initialize VB1 and VB2

repeat

<event-gn.aph level pha6e>

match. event-graph base with VB1 and determine enabled event gnaphs

resolve conflict

match selected event-graphs with VB1 and determine active, event-graphs

*re*olve conflict*

fire all active events and exacute relevant actions on DBI and DB2

*<rule level pha*e>*

*match rule ba*e with DB2 and determine conflict set*

resolve conflict

execute selected rule/s on VB2 and VB1

until goal condition

Figure 3 - Basic mode of operation of the inference engine

The architecture above sketched is very general. Both levels can contain representation of domain knowledge and meta-knowledge, and both DBI and DB2 can embody knowledge on the problem and control knowledge on the problem solving strategy. The rule base can be organized according to any of the usual structuring techniques, such as partitioning, meta-rules, etc. Also the event-graph base can be given arbitrarily complex structure by defining relations among EGs.

Both the event-graph and rule levels maintain in their own data base (DBI and DB2 respectively) the information needed for their operation (computation and deduction respectively). Whenever some information has to be shared by the two levels it must be duplicated in the two data bases.

Each of the two levels can exert control on the other: rules can influence the operations of event-graphs by forcing, conditioning, or suspending their activation through modifications of the current markings stored in DBI; on the other hand, EGs actions can change the conditions for matching rules by modifying the content of DB2.

The novel expert system paradigm above introduced embodies the features of usual rule-based paradigm as well as the ones proper of imperative programming languages. The most relevant contribution of our approach is the definition of an environment where these two components can naturally cooperate in a problem solving task. In this respect, it shares some characteristics with the work of Georgeff and Bonollo (1963), but it also includes several new points:

the visibility of the system on pieces of procedural knowledge is not limited to input/output information (procedural abstraction); specific mechanisms are provided to access the internal state and influence the dynamic behaviour of EGs;

the design of the right dimension of pieces of procedural knowledge is not a critical point as intermediate results of partial EG computations can be accessed and used (note that generally too small fragments tend to reduce the advantages of having explicit representation of procedural knowledge, while too large chunks imply the possibility of long, useless computations, as once a procedural area is entered its interruption causes the loss of all intermediate results);

representation and use of declarative knowledge is not limited to the invocation part of a knowledge area.

PROBLEM ANALYSTS AND SYSTEM SPECIFICATIONS

In this section we focus on the problem of on-line monitoring of water pollution in a thermal power plant that has been used as a testbed for the new expert system architecture introduced in the previous section.

The usefulness of artificial intelligence techniques for assisting and advising the operator of a power plant in case of accidents has already been stressed in (Nelson 1982) and (Underwood 19B2). Our problem, even if it is not concerned with plant safety, lies in this application area. The aim of the system is to avoid damages to critical subsystems of the power plant and to limit plant unavailability caused by chemical agents, by means of early fault detection and diagnosis, and appropriate intervention.

A simplified schema of a thermal power plant is given in Figure 4. In order to ensure the correct operation of the plant and to preserve the materials constituting some critical subparts, the cycle water must be kept as pure as possible. In fact, possible pollutants dissolved in the cycle water

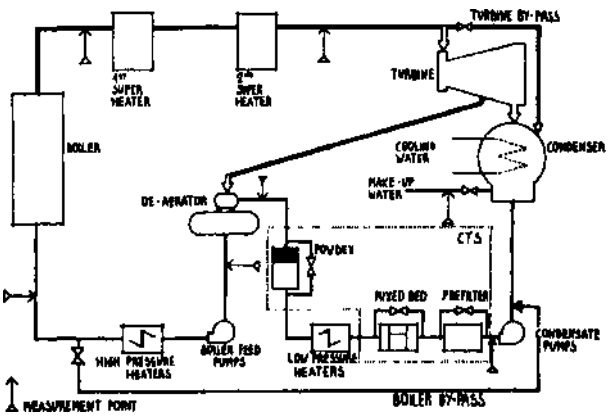

Figure 4 - Semplifled schema of a thermal power plant

may react with the plant in different ways, mainly at high temperature, giving both oxidation products and acids that can damage the tubes, the walls of the boiler, and the rotating parts of the turbine.

Therefore a chemical treatment subsystem (CTS) is included in the plant and is used by a human operator to clean the cycle water whenever it is affected by some pollutant. The CTS includes two different kinds of filters:

- the mixed bed, containing ion exchange resins, that operates a chemical filtering;
- the pre filter and the powdex, containing cellulose and pulverized resin respectively, that remove oxides through a physical filtering.

The insertion level of the filters is regulated by the operator depending on the nature and quantity of the pollutants in the water.

Pollutants of different kinds can enter into the cycle:

- marine or river water carrying salts and sand (through the condenser);
- air (in particular, oxygen and carbonic dioxide);
- cellulose and resins released by the CTS itself.

Pollution phenomena are controlled by continuous measuring of several chemical parameters (total conductivity, acid conductivity, oxygen concentration, etc.) at several points of the cycle (see Figure 4).

The control equipment currently installed in the plant can automatically open the valves that by-pass the critical subsystems (e.g., boiler or turbine) when one of the above mentioned chemical parameters at a single measurement point increase beyond a fixed threshold level, in order to avoid immediate damage. The task of the operator is to prevent the pollution reaching this limit situation by properly activating the CTS as soon as some sensors show abnormal values. In fact, when a limit situation occurs, some components of the plant may have already been damaged, and, in any case, the by-pass of a critical subsystem causes a significant loss of power.

During working time the chemical staff can advise the operator on the most appropriate actions to be taken to control an incipient pollution phenomenon; during the night, week-end, and holidays the operator must follow the instructions in a handbook.

The operative procedure above described has often proved to be unsatisfactory for several reasons:

- The operator usually guesses that a pollution phenomenon is occurring only when the sensors show highly abnormal values, so that the corrective action is not as timely as it should be.
- Sometimes, the presence of a pollutant can be detected only by correlating the temporal trends of different parameters. These correlations require expertise not available to the operator.
- The operator is inclined to reason in terms of short time effects. Therefore, he often underestimates slight pollutions that, however, can generate long term damage.
- When the operator has to manage situations without the advice of the chemical staff, he is often in trouble as the procedures specified in the handbook are too simple and schematic and turn out to be be useless in complex cases.

In order to face these problems, we have investigated the possibility of using a computer-based system that can acquire chemical parameters from the sensors on the plant and give advice to the operator. A preliminary analysis of the knowledge required to recognize abnormal situations, identify the kind of pollutant, and suggest the proper intervention has shown that expert system methodology can be more advantageous than traditional approaches.

This led to the design and implementation of *PROP,* a real time expert system that can assist the CTS operator. PROP directly acquires data from the plant through sensors at a fixed time rate. These measurements concern general information on the plant (e.g., flow rate, etc.), chemical parameters (e.g., acid conductivity, oxygen concentration, etc.), and states of some subsystems (e.g., mixed bed insertion, de-aerator valve position, etc.). Moreover, through the keyboard, the operator can introduce further information that cannot be directly acquired by sensors (e.g., results of off-line chemical analyses, etc.).

As soon as an anomaly is detected, PROP displays a set of hypotheses on the operator console, that can explain the malfunctioning. Afterwards, the system keeps the evolving situation under control, tries to focus on the most plausible hypothesis, and suggests proper interventions.

After the situation has been brought to normality, PROP closes the diagnosis/intervention session, informs the operator as to the success of the undertaken interventions, and resumes normal monitoring activity.

KNOWLEDGE REPRESENTATION

Expert knowledge involved in the problem illustrated in the previous section can be organized at three different levels. The human expert uses, in fact, three kinds of knowledge: each of them is Involved in a particular class of activities.

Data interpretation level

It comprises the techniques that make the expert able to collect and analyze information about the behaviour of a large number of chemical and state parameters and to derive a representation of the state of the plant from the point of view of cycle water pollution. In other words, it contains knowledge about how to observe the plant in order to translate parameter values and trends into symbolic, higher level conceptual representation.

A data interpretation activity is, for example, the early perception of an alarm condition detected by correlating a fast power decrease with a smooth rise, after a definite time interval, of the acid conductivity measured before the de-aerator.

Such reasoning can be carried out only by paying attention to a restricted set of the available signals and parameters, with a precise concept of what one is looking for. The human expert is able to carry out a rather large number of different data interpretation activities and to choose the right one taking into account the current state of his hypotheses and expectations.

Diagnosis /intervention level

It comprises the techniques currently utilized for fault diagnosis, i.e. for recognizing one or more of the possible pollution causes and for performing appropriate recovery actions. Each diagnosis/intervention activity describes the hypotheses, expectations, and actions that relate to a specific pollution situation. It specifies which data interpretation activities have to be started, and how their results have to be evaluated in order to validate (or reject) a certain diagnosis.

The confirmation of a suspected condenser leakage and the control of the leakage propagation along the plant are, for example, the tasks of two diagnosis/intervention activities.

Control level

When an abnormal situation is detected in the plant, the expert formulates a number of hypothetical diagnoses. Each of them can be validated or rejected by means of a diagnosis/intervention activity, and all of these tasks

must be carried on in parallel. In a given instant the state of each diagnosis/intervention activity and the results, though partial, of each related data interpretation activity give an image of the state of the plant from the point of view of a hypothetical diagnosis.

Before a single diagnosis is definitely ascertained or when concurrent pollutions take place, the expert has to manage relationships and conflicts among all the active diagnosis/intervention activities. Moreover, successful results achieved by a specific diagnosis/intervention activity are often detrimental to other ones. In general, in dealing with complex situations, chunks of information concerning the state of several diagnosis/intervention activities are to be compared.

For example, a diagnosis/intervention activity which is concerned with propagation of a condenser leakage must inhibit consideration of all the diagnosis/intervention activities that assume a fault in a subsystem that is located further down the condenser in the cycle (e.g., powdex, de-aerator, etc.).

The human expert carries out this crucial control task by means of strategic knowledge, that makes him able to activate, deactivate, privilege, and influence the diagnosis/intervention activities necessary to globally diagnose and control the behaviour of the plant.

Data interpretation and diagnosis/intervention activities can be easily expressed in a procedural language. They can, in fact, be naturally seen as pieces of procedural knowledge:

data interpretation activities are procedures that specify a sequence (with selections and iterations) of elementary observations of the plant;

diagnosis/intervention activities are higher-level procedures that comprise as elementary steps the invocations and consultations of a number of data interpretation activities.

A main point is that the dynamic behaviour of such procedures must be easily observed and influenced from outside. Procedures are important here not only for the results they can compute, but specially for the intermediate computations they perform. The diagnosis/intervention level works on the behaviour of the data interpretation activities, and the control level must exert its influence on both diagnosis/intervention and data interpretation activities.

Expert knowledge that deals with time, which is of crucial importance in on-line applications, has an explicit role in our application at the data interpretation level only. In fact, experts explicitly refer to time relations only when they describe the procedures they use for observing the behaviour of the plant. The procedural nature of data interpretation level knowledge makes it easy to represent such temporal statements explicitly as time interval measurements.

Control knowledge is more declarative and fragmentary in nature. It can be represented by means of production rules to be used in a non-deterministic way.

The knowledge analysis above developed can be usefully compared to the approaches of Nii, Feigenbaum, Anton, and Rockmore (1982) on HASP/SIAP, of Kahn (1982) on MUD, and of Hudlicka and Lesser (1984) on FDD.

We believe that the model above introduced for the analysis of knowledge involved in the diagnosis of cycle water pollution of a thermal power plant embodies several features of general significance. It could easily be applied to a large class of tasks concerning real-time fault detection, diagnosis, and intervention in complex systems.

PROP ARCHITECTURE

Now we can merge the knowledge analysis above developed with the general architecture proposed.

The procedural knowledge of the data interpretation and diagnosis/intervention levels can be properly represented in the event graph-base as a collection of independent EGs. Though data interpretation and diagnosis/intervention EGs have the same syntactic representation, their events are labeled with expressions of two different languages:

Expressions appearing in data interpretation EGs must be evaluated with respect to chemical and state parameters acquired from the sensors (stored in DBI). or with respect to time intervals. A condition referring to a time interval starts a counter the first time it is evaluated and it remains false until the time interval has expired.

Expressions of diagnosis/intervention EGs, on the other hand, refer to the markings of data interpretation EGs.

Current markings of EGs are represented in DBI and duplicated in DB2 inuch a way as to be explicitly and directly accessible by the rule level .too. The evaluation of a condition bound to an event of a diagnosis/intervention EG that refers to a not yet enabled data interpretation EG causes its enabling, i.e. the places in its initial marking are marked. Therefore, a data interpretation EG has a current marking for each diagnosis /intervention EG that uses it. This represents the capability of the system to exploit, in an independent way, a single data Interpretation procedure in different contexts.

Both EG base and DBI are naturally partitioned. In fact, it must be possible to distinguish data interpretation EGs from diagnosis /intervention ones in the EG base and each of the independent markings of the data interpretation EGs in DBI.

Control level knowledge fits in the rule base. It is represented by production rules whose conditions and actions refer to markings of both data interpretation and diagnosis/intervention EGs.

Moreover, a *credibility value* is defined for each active diagnosis/intervention EG. It represents the promise for the EG to be most appropriate, up to the moment, for explaining the ongoing phenomenon and for suggesting the right actions on the plant. The maintenance of the credibility values is provided by the control level. They are used in conflict resolution activities.

As EG markings and credibility values seem to be the only information required by the control level in our particular applicatioa we have merged in the implementation DBI and DB2 into a single data base. This simplifies and speeds up system operation without loss of power, In fact, as pointed out above, in our application declarative knowledge has a controlling role only.

AN EXAMPLE

PROP has been implemented in Franz Lsp on a SUN-2 workstation at CISE laboratories and it has been tested on recorded patterns comprising a wide variety of different operating situations. A pilot on-site installation of PROP is presently being developed.

This section illustrates a few examples of PROP operation. For easier understanding the examples are reported in a simplified way.

The examples refer to the diagnosis of pollution arising in the condenser. To detect this kind of anomaly as soon as it occurs, the human expert must focus on the set of chemical parameters measured around the condenser. In

particular, the increase of the acid conductivity between the condenser inlet and outlet must be considered a significant event. Therefore a data Interpretation activity (CONDENSER-CONTROL) concerning chemical parameter trends in the condenser has been defined. Acid conductivity increase beyond the normal level is an elementary observation included in this data interpretation activity. When such an event occurs, the human expert realizes that a pollution from the condenser is likely to take place, and he immediately defines some objectives that will support his successive observations and suggests proper corrective actions. The main objectives considered in this situation are reported below: each of them is represented in PROP by a diagnosis/intervention activity (mentioned in parentheses):

control of the pollution trend in relation to the intervention undertaken to remove the cause of pollution (CONDENSER-LEAKAGE);

control of the pollution propagation along the plant (in particular towards the boiler) and reduction of the pollution effects by means of insertion of CTS filters (CONDENSER-LEAKAGE-PROPAGATION);

detection of the kind and source of pollution. In fact, pollution can be due to cooling water (COOLING-WATER-LEAKAGE), to air contamination from condensate pumps (AIR-CONTAMINATION), or to impurities of make-up water (POLLUTANT-FROM-MAKE-UP-WATER).

Figure 5a illustrates the EG representing the CONDENSER-LEAKAGE diagnosis/intervention activity. Events are labelled here with conditions only. The general form of a condition is a logical expression involving predicates of the type: (<DATA-INT-ACT-NAME> in <PLACE-NAME>), that is true if, at evaluation time, the place <PLACE-NAME> in the data interpretation EG <DATA-INT-ACT-NAME> is marked.
When the CONDENSER-CONTROL data interpretation activity has a mark in the place LEAKAGE-DETECTION, the places in initial marking (FIRST-RISE) of the CONDENSER-LEAKAGE EG are marked.

The evolution of the diagnosis/intervention activity according to the CONDENSER-LEAKAGE EG depends on the following events:

(1) The ratio between acid conductivities at condenser inlet and outlet becomes stable (STEADY). In this case the pollution is assumed to have reached a defined value.

(2) The ratio between acid conductivities at condenser inlet and outlet shows a peak, i.e., the acid conductivity suddenly increases and then rapidly comes back to normal values (NORMALITY).

(3) The acid conductivity keeps a steady value for a while and then decreases as a result of an appropriate intervention, but without reaching a normal value (DECREMENTED).

(4) Pollution comes back to normal values (NORMALITY).

(5) Pollution increases again (NEW-RISE).

When the place FIRST-RISE of the CONDENSER-LEAKAGE EG is marked, events E1 and E2 are enabled. The data interpretation activities CONDENSATE-PAR-TREND and CONDENSATE-PAR-VALUE are started. Event El will fire when the CONDENSATE-PAR-TREND EG has the AC0ND1-STEADY place marked and event E2 has not yet fired.

In Figure 5b we show the the CONDENSATE-PAR-TREND EG. This EG verifies whether the acid conductivity increase measured at the condenser outlet has reached an abnormal value Moreover, it checks the oxygen concentration variation during the conductivity increase. As mentioned above, this data interpretation activity is used to verify the condition c1 of event El in the CONDENSER-LEAKAGE EG. Other information given by this EG (e.g., check of the oxygen increase) is used by control level rules (RULE-29 - see below).

Predicates of data interpretation EGs are evaluated with respect to an object oriented organization of the acquisition buffers and of the low level numerical programs,

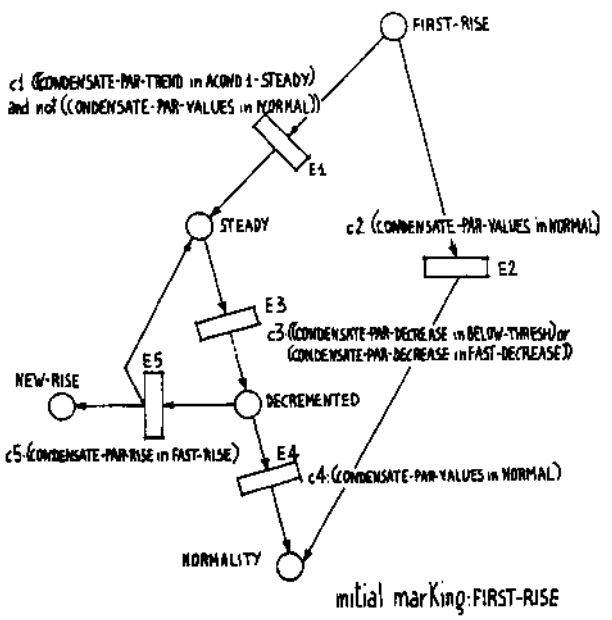

Figure 5a - CONDENSATE-LEAKAGE event-graph

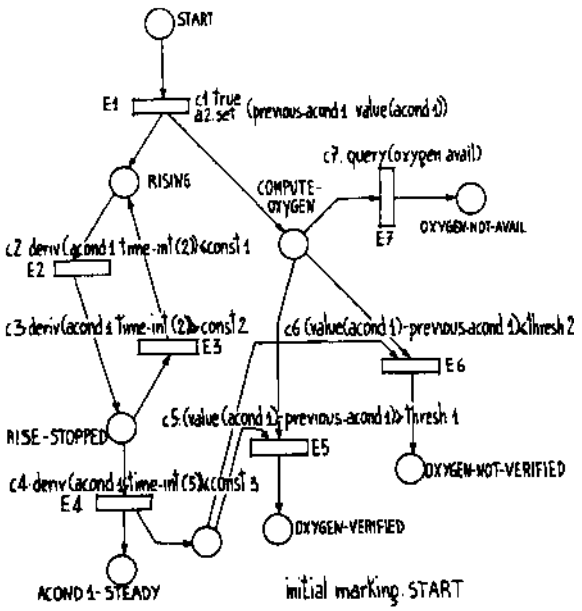

Figure 5b - CONDENSATE-PAR-TREND event-graph

i.e. average and derivative calculations, etc. This choice was made to increase system flexibility in the interface with either a possible simulation environment or an operational one. A complete description of this representation schema is beyond the aim of this paper, so that a rather free predicate calculus-like notation is adopted in the EG of Figure 5b. Most of the expressions that label events in the CONDENSATE-PAR-TREND EG are self-explanatory. In addition, we point out the following:

An action (setting the local variable 'previous-acondl') is associated with event E1. In the other events the action field is empty.

In events E2, E3, and E4, the evaluation of the predicates c2, c3 and c4 involves a call to a numerical program that computes the average slope on a specified time interval.

In events E5 and E6 the predicates c5 and c6 specify the difference between values of a parameter in two different instants.

The function "query", in event E7 means that PROP will ask to the human operator the values necessary to compute the truth value of condition c7.

Some examples of rules defined at the control level are shown in Figure 6.

In the action side of RULE-29 the enabling of two diagnosis/intervention activities is defined These diagnosis/intervention activities are enabled in order to investigate the pollution source: the former (AIR-FROM-CONDENSER) verifies whether air contamination (for example from a condensate pump) is present, the latter (POLLUTANT-FROM-MAKE-UP-WATER) considers impurities coming from the make-up water

RULE-29

±1

CODENSER-LEAKAGE in STEAVV and

CONDENSATE-PAR-TERNDS

invoked-by CONDENSER-LEAKAGE

in OXYGEN-VERIFIED

then

enable AIR-FROM-*CONDENSER*

enable POLLUTANT-FROM-MAKE-UP-WATER

RULE-55

ii

CONDENSER-LEAKAGE IN ACTIVE *and*

POLLUTANT-FROM-MAKE-UP-WATER in WAIT-CONFIR-MATION *and*

AIR-FROM-CONPENSER i*n V*ERIFIED

then

deactivate. POLLUTANT-PROM-MAKE-UP-WATER

Figure 6 - Examples of rules

The premise of RULE-29 is satisfied if;

the CONDENSER-LEAKAGE diagnosis/intervention aotivity has been enabled and it verifies that the pollution level has stopped increasing, that is, its EG has the place STEADY marked;

the CONDENSATE-PAR-TRENDS data interpretation activity activated by the CONDENSER-LEAKAGE diagnosis/intervention activity verifies an increase of oxygen concentration.

Note that the two diagnosis/intervention activities enabled by RULE-29 are concurrently involved in the explanation of the same phenomenon. The management of the results obtained from these two different points of view is accomplished by means of other control level rules, with the aim of definitely validating one hypothesis and rejecting the other.

For example, RULE-55 disregards the hypothesis that the pollution is due to the make-up water (deactivate POLLUTANT-FROM-MAKE-UP-WATER) because, in the mean time, the hypothesis of air contamination from the condenser (AIR-FROM-CONDENSER) has been verified (its VERIFIED place is marked). This takes place even if the POLLUTANT-FROM-MAKE-UP-WATER EG cannot yet give definite results as it is still waiting for the firing of some of its events.

CONCLUSIONS

The work done with PROP has allowed definition and testing of a novel rule-based system architecture supporting heterogeneous knowledge representation. The proposed approach is expected to offer three major advantages:

the knowledge representation mechanism constituted by the fusion of production rules and event-graphs makes knowledge acquisition more effective and easy and knowledge organization more transparent and natural;

non-determinism is limited to those aspect of the problem domain which actually imply non-deterministic operation, and is not used extensively thus overcoming fictitious search problems deriving from a poor knowledge representation schema;

explanation capability can closely follow the path of human reasoning, as a consequence both of the naturalness of knowledge representation and of the constrained non-determinism of the inference process.

The research has disclosed several new issues to be considered in future work. Among these we stress the extension of the proposed architecture to include explicit representation of the communication net connecting heterogeneous knowledge chunks (Davis and Smith, 19B3), thus overcoming the limitations of the simple communication schema currently adopted. Moreover, explicit temporal reasoning, credibility values computation and propagation, and the implications of the proposed architecture on meta-level structure are going to be investigated.

ACKNOWLEDGMENTS

The construction of the knowledge base of PROP has been possible thanks to the experts of several ENEL Departments (Construction, Operation and Research & Development Departments) who have cooperated with CISE in the frame of an internal ENEL working group. We are indebted to the above Departments for the support and contribution offered. In particular, we would like to thank Giovanni Quadri of ENEL-DCO, Flavio Bacci of ENEL-DPT, and Raffaele Pascali of ENEL-DSR.

REFERENCES

Buchanan, B.G. and Duda, R.O. (1983). Principles of Rule-Based Expert Systems. In M.C. Yovits (Ed.), *Advances in Computers* 22, Academic Press, New York, NY, 163-218.

Davis R. (1980a). Meta-Rules: Reasoning about Control. *Artificial Intelligence* 15, 179-222.

Davis R. (1980b). Content Reference: Reasoning about Rules. *Artificial Intelligence* 15, 223-239.

Davis R. and Smith R.G. (1983). Negotiation as a Metaphor For Distributed Problem Solving. *Artificial Intelligence* 20, 63-109.

Friedland P. (1981). Acquisition of procedural knowledge from domain expert. *Proc. 7th Int. Joint Conf. on Artificial Intelligence,* Vancouver, BC, Canada, 856-861.

Genesereth M. (1983). An Overview of Meta-level Architecture. *Proc. 3rd Nat. Conf. on Artificial Intelligence,* Washing toa DC, 119-124.

Georgeff MP. (1982). Procedural Control in Production Systems. *Artificial Intelligence* 18, 175-201.

GeorgefT M.P. and Bonoilo U. (1983). Procedural Expert Systems. *Proc. 8th Int. Joint Cbnf. on Artificial Intelligence,* Karlsruhe, FRG, 151-157.

D'Angelo A., Guida G.. Pighin M., and Tasso C. (1985). A Mechanism for Representing and Using Meta-knowiedge in Rule-Based Systems. In M.M. Gupta, A. Kandel, W. Bandler and J.B. Kiszka (Eds), *Approximate Reasoning in Expert Systems,* North-Holland, Amsterdam, NL (to appear).

Hayes-Roth F., Waterman DA, and Lenat D. (Eds.) (1983). *Building Expert Systems,* Addison-Welsey, Reading, MA.

Hudlicka A. and Lesser V. (1984). Meta-Levei Control Through Fault Detection and Diagnosis. *Proc. 4th Nat. Conf. on Artificial Intelligence,* Austin, TX, 153-161.

Kahn G. (1984). On When Diagnostic Systems Want To Do Without Causal Knowledge. In T. O'Shea (Ed), *ECAI-84: Advances in Artificial Intelligence,* Elsevier Science, Amsterdam, NL, 21-30.

Nelson W. (1982). REACTOR: An Expert System For Diagnosis and Treatment of Nuclear Reactor Accidents. *Proc. 2th Nat, Cbnf. on Artificial Intelligence,* Pittsburgh, PA, 296-301.

Nii H.P., Feigenbaum E.A., Anton J.J., and Rockmore A.J. (1982). Signal-to-Symbol Transformation: HASP/SIAP Case Study. *The AI Magazine* 3(1), 23-36

Reisig W. (19B2). *Petrunetze,* Springer-Verlag, Berlin, FRG.

Underwood W.E. (1982). A CSA Model-Based Nuclear Power Plant Consultant. *Proc. 2th Nat. Conf. on Artificial Intelligence,* Pittsburgh, PA, 302-305.